App-based Forecasting of CRIX Index Returns Using R and R-Shiny

Master's Thesis submitted

to

Prof. Dr. Wolfgang Härdle

Prof. Dr. Brenda López Cabrera

Humboldt-Universität zu Berlin
School of Business and Economics
Institute for Statistics and Econometrics

by

Gonzalo Agustin Garcia Camargo
(598851)

in partial fulfillment of the requirements
for the degree of
Master of Science in Statistics
Berlin, June 28, 2021

Acknowledgement

I would like to thank Prof. Dr. Wolfgang Härdle for being a constant source of inspiration, for his guidance and support while writing this thesis and also for the many insights I gained while attending his classes.

I would also like to thank Prof. Dr. Brenda López Cabrera for her support and advice while writing this thesis and the valuable lessons learned on her Statistical Tools in Finance and Insurance course.

Finally, I would like to thank my family and specially my wife and my child for their unflinching support. They provide me with the motivation to try and keep bettering myself.

Abstract

This thesis aims to deliver a model to beat a Naive and Mean Benchmark at predicting the Log Returns of the CRIX (CRyptocurrency IndeX) on a one day forecasting horizon. Various models were tried and tested for their performance, finally settling on an LSTM model. This model was able to beat both the Naive and Mean Benchmarks on Cross-Validation Mean Absolute Error. It also performed well on held-out data which was not used for training or model selection. A secundary objective of this thesis was to integrate the results of the investigation on a Web App based on R-Shiny to facilitate the usage of the trained model. All observations from the history of the CRIX model were used for training and testing the models (2504 obs. as of June 7th, 2021). The App can be found at `https://github.com/QuantLet/CRIXForecastApp`.

Contents

List of Abbreviations

GRU	Gated Recurrent Unit	RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory	ETS	Error, Trend, Seasonality
BTC	Bitcoin	ETH	Ethereum
BNB	Binance Coin	XRP	Ripple
DOGE	Dogecoin	USDT	Tether
ADA	Cardano	DOT	Polkadot
LTC	Litecoin	LINK	Chainlink
CV	Cross-Validation	USD	United States Dollar
PoW	Proof-of-Work	PoS	Proof-of-Stake
CC	Cryptocurrencies	App	Application
ACF	Autocorrelation Function	CRIX	Cryptocurrency Index
ML	Machine Learning	MAE	Mean Absolute Error
DAX	Deutscher Aktienindex	BPTT	Backpropagation Through Time
VCRIX	Volatility Index for CC		
PACF	Partial Autocorrelation Function	ARIMA	Autoregressive Moving Integrated Average

List of Figures

List of Tables

1 Introduction

On the 1st of November 2008 a group or person under the alias "Satoshi Nakamoto" published a paper laying the foundations for a trust-less system for electronic transactions (Nakamoto, 2008). This paper explained the Bitcoin-Network, which was born two months later on the 3rd January 2009 with the creation of Block 0 and the first 50 Bitcoins (BTC) (Blockchain.com, 2021)

Previous digital coins had faced the issue of preventing someone from spending the same coin twice (double-spending), without having a central authority to monitor transactions. Nakamoto solved the double-spending problem through the use of a decentralized network implementing proof-of-work (PoW), which rendered double spending impracticable under some easy to achieve conditions and with little to no coordination.

Bitcoin is only one of a whole suite of Blockchain solutions with a wide range of applications such as providing a safe haven from inflation (Blau et al., 2021), making traditional financial services available without the need of intermediaries, cloud storage, decentralization of organizational structures, E-Government, Proof of ownership, etc (Buterin, 2013). Many of this solutions have Cryptocurrencies associated to them, examples of this being Ether for the Ethereum Network or Bitcoin for the Bitcoin Network.

Furthermore, over the last decade Cryptocurrencies have taken hold as one of the world's most dynamic and interesting assets, with Bitcoin reaching a value of over $64800 per coin and a Market Capitalization of over $1,179,061,093,980 (CoinMarketCap, 2021a) and Ethereum reaching a value of over $4300 and a Market Cap of $438,585,075,674 (CoinMarketCap, 2021b), capturing not only the attention of cryptographers and geeks, but also of the financial and political elites of our world.

Many of the world's big technology companies are also looking into the advantages cryptocurrencies can offer. An example of this would be Diem (previously called Li-

bra), an association which includes companies such as Facebook, Spotify and Uber as members (Diem, 2021) and whose purpose is to create a global payments system and financial infrastructure with the goal of fostering financial inclusion (Diem, 2019). With many more heavy weights like Mastercard, PayPal and Ebay reported to initially support the agreement which have either left the association, are ambiguous about the project or are refusing to comment on their position highlighting that Blockchain can also be controversial (Business Insider, 2021).

The attention Blockchain technology has received is not only of the positive kind. Governments around the world are looking at regulating the Cryptocurrency space and also getting in on the action themselves. Concerns about crime, sovereignty and opportunity are the drivers behind this current governmental interest on the Blockchain (See European Central Bank (2020), European Commission and European Central Bank (2021) and Cheng et al. (2021)).

Moreover, the Bitcoin Network depends on PoW and PoW/PoS (proof-of-stake) hybrid schemes which rely on costly computations to ensure the good functioning of the network. This, however leads to massive energy consumption, with the Bitcoin Network's annual consumption being compared to that of some small countries (Badea and Mungiu-Pupãzan, 2021). With governments moving to present a united front against climate change (United Nations, 2015), the energy consumption of PoW might provide Bitcoin's detractors with an argument against it. Luckily, there are alternative consensus mechanisms like PoS, which have a much smaller energy consumption than PoW schemes, even though they are not as secure (Vranken, 2017).

It is therefore of the utmost importance that, as the Cryptocurrency market gains in size, relevance and disruptive potential, we become able to better understand the behavior of this market. Many efforts have been made in this direction. One example would be the CRyptocurrency IndeX (CRIX) from Trimborn and Härdle (2018), around which many other tools have been and continue to be built. Some examples of that are the VCRIX, a volatility index based on the CRIX (Kim et al., 2019) and a methodology for BTC option pricing (Chen et al., 2018). The CRIX and VCRIX can be found at

(https://thecrix.de/).

In this paper an attempt is made at trying to better comprehend and forecast the future behavior of the CRIX. In order to do this, a LSTM Neural Network was trained with the goal of beating both naive and mean-average baselines on a one-day forecast of the log returns of the CRIX Index. Furthermore, this has been integrated into a web application in order to make it accessible to anyone. The application (from here on App) is based on R Shiny as well as some of the best deep learning libraries available on the R programming language. It is supposed to be easy to use and comprehend. It can be found online at https://github.com/QuantLet/CRIXForecastApp.

The paper is organized as follows. Section 2 describes the models which were used, either as baselines or to actually attempt a good forecasting performance. Section 3 describes the data set, as well as the implementation of the Web App. Section 4 presents the results. Finally, the conclusion can be found in Section 5.

2 Models

2.1 Naive Model

The Naive model is based on simple reasoning. If we were asked to forecast how much the expenses of our household are going to be on the next month, the answer with lowest complexity (and lowest cost in money or effort), while still being reasonable, would be: "They will be similar to the expenses of this month". If we were asked for an Euro-amount, we might as well name the exact same value as was expended this month. This is the naive model. In short, our forecast for a variable y on period $T+1$ is going to be the value of y on T. That is:

$$\hat{y_{T+1}} = y_T \tag{1}$$

Due to its simplicity, the Naive model can be used as a benchmark against which to measure the performance of other forecasting models. This is the reason it is included among the considered models.

2.2 Mean Model

Given we will be dealing with log returns, we will not have a series that constantly grows, but rather one that oscillates around zero. If we assumed that the series was a white noise, then it's expected value of zero would be just as good a predictor as any other. If the series is more often bigger than zero than it is smaller, we know it is not a white noise, but it might still be quite difficult to predict.

Without using an elaborate model, but trying for a better solution than the Naive one, we might come to the idea of using the mean of all past observations as our prediction value. For sure we will not be close on each individual prediction, but on average we will be close to the actual value of the series. In short, the mean model can

be mathematically expressed as follows:

$$\hat{y_{T+1}} = \frac{\sum_{i=1}^{T} y_i}{T} \tag{2}$$

2.3 ETS Model

$ETS(.,.,.)$ stands for Error, Trend , Seasonality. It is a framework for exponential smoothing methods (Hyndman et al., 2002). The types of Trend and Seasonality we find are None (N), Additive (A) and Multiplicative (M). Additionally, we have Additive Damped (A_d) and Multiplicative Damped (M_d) Trend. For example, the Holt-Winters' (Winters, 1960) deterministic method with additive trend and multiplicative seasonality would be an $ETS(N, A, M)$.

This model performs well on short-term forecasts as the one being attempted (Hyndman et al., 2002). The 12 resulting configurations can be written as follows:

$$\begin{aligned} l_t &= \alpha P_t + (1-\alpha) Q_t, \\ b_t &= \beta R_t + (\phi - \beta) b_{t-1}, \\ s_t &= \gamma T_t + (1-\gamma) s_{t-m} \end{aligned}$$

with l_t, b_t, s_t respectively the level, the slope and the seasonal component of the series at time t. α, β, γ and ϕ are parameters to be estimated. Finally, P_t, Q_t, R_t and T_t determine which configuration of the model is used (see Figure 1).

Trend component	Seasonal component: N (none)	A (additive)	M (multiplicative)
N (none)	$P_t = Y_t$ $Q_t = \ell_{t-1}$ $\phi = 1$ $F_{t+h} = \ell_t$	$P_t = Y_t - s_{t-m}$ $Q_t = \ell_{t-1}$ $T_t = Y_t - Q_t$ $\phi = 1$ $F_{t+h} = \ell_t + s_{t+h-m}$	$P_t = Y_t / s_{t-m}$ $Q_t = \ell_{t-1}$ $T_t = Y_t / Q_t$ $\phi = 1$ $F_{t+h} = \ell_t s_{t+h-m}$
A (additive)	$P_t = Y_t$ $Q_t = \ell_{t-1} + b_{t-1}$ $R_t = \ell_t - \ell_{t-1}$ $\phi = 1$ $F_{t+h} = \ell_t + hb_t$	$P_t = Y_t - s_{t-m}$ $Q_t = \ell_{t-1} + b_{t-1}$ $R_t = \ell_t - \ell_{t-1}$ $T_t = Y_t - Q_t$ $\phi = 1$ $F_{t+h} = \ell_t + hb_t + s_{t+h-m}$	$P_t = Y_t / s_{t-m}$ $Q_t = \ell_{t-1} + b_{t-1}$ $R_t = \ell_t - \ell_{t-1}$ $T_t = Y_t / Q_t$ $\phi = 1$ $F_{t+h} = (\ell_t + hb_t) s_{t+h-m}$
M (multiplicative)	$P_t = Y_t$ $Q_t = \ell_{t-1} b_{t-1}$ $R_t = \ell_t / \ell_{t-1}$ $\phi = 1$ $F_{t+h} = \ell_t b_t^h$	$P_t = Y_t - s_{t-m}$ $Q_t = \ell_{t-1} b_{t-1}$ $R_t = \ell_t / \ell_{t-1}$ $T_t = Y_t - Q_t$ $\phi = 1$ $F_{t+h} = \ell_t b_t^h + s_{t+h-m}$	$P_t = Y_t / s_{t-m}$ $Q_t = \ell_{t-1} b_{t-1}$ $R_t = \ell_t / \ell_{t-1}$ $T_t = Y_t / Q_t$ $\phi = 1$ $F_{t+h} = \ell_t b_t^h s_{t+h-m}$
D (damped)	$P_t = Y_t$ $Q_t = \ell_{t-1} + b_{t-1}$ $R_t = \ell_t - \ell_{t-1}$ $\beta < \phi < 1$ $F_{t+h} = \ell_t +$ $(1 + \phi + \cdots + \phi^{h-1}) b_t$	$P_t = Y_t - s_{t-m}$ $Q_t = \ell_{t-1} + b_{t-1}$ $R_t = \ell_t - \ell_{t-1}$ $T_t = Y_t - Q_t$ $\beta < \phi < 1$ $F_{t+h} = \ell_t +$ $(1 + \phi + \cdots + \phi^{h-1}) b_t + s_{t+h-m}$	$P_t = Y_t / s_{t-m}$ $Q_t = \ell_{t-1} + b_{t-1}$ $R_t = \ell_t - \ell_{t-1}$ $T_t = Y_t / Q_t$ $\beta < \phi < 1$ $F_{t+h} = [\ell_t +$ $(1 + \phi + \cdots + \phi^{h-1}) b_t] s_{t+h-m}$

Figure 1: Formulae for different ETS Model Configurations from (Hyndman et al., 2002)

2.4 LSTM Networks

2.4.1 RNNs

A Recurrent Neural Network (RNNs) consists of a hidden state and an output y. It can learn to predict sequence x by learning the distribution of the data over the sequence (Cho et al., 2014). Its mathematical expression is the following (Yu et al., 2019):

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b),$$
$$y_t = h_t$$

where W_h and W_h are the weights, b the bias, x_t is the input, h_t the recurrent information and y_t the output at time t. See Figure 2 for a graphical representation of a RNN unit.

In theory, RNNs should be easy to train using Backpropagation Through Time (BPTT) (Werbos, 1990). However, their training actually tends to be problematic because of the vanishing/exploding gradient problem exposed on the next section. This has stopped this basic form of the RNN from becoming part of the mainstream Machine Learning tool set (Sutskever et al., 2011).

2.4.2 RNNs and the vanishing and exploding gradient problem

As illustrated by (Goodfellow et al., 2016), RNNs use the same matrix $\mathbf{W}$ for each time-step while updating. This implies multiplying by $\mathbf{W^t}$ where t is the number of time-steps the back-propagation algorithm has gone through. Let us assume that the eigendecomposition of $\mathbf{W}$ is as follows:

$$W = (V diag(\lambda) V^{-1})^t = V diag(\lambda)^t V^{-1}$$

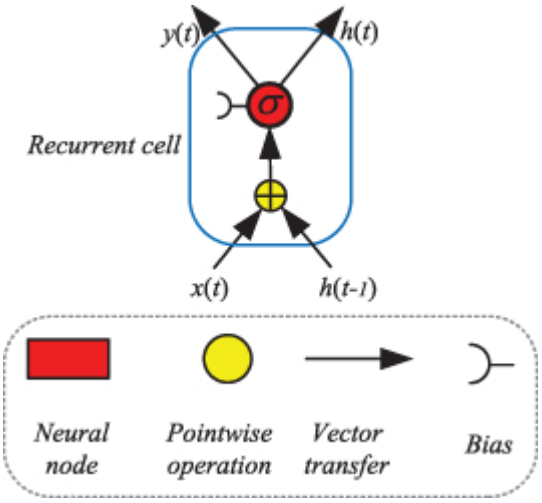

Figure 2: RNN cell from Yu et al. (2019)

Then the gradients get scaled according to $diag(\lambda)^t$ causing any that are sufficiently smaller than 1 to vanish and any that are sufficiently over 1 to explode. This is the reason why RNNs have problems learning when the relevant inputs to the outputs lie more than 5-10 time steps apart (Gers et al., 2000).

It is possible to train Deep Feed-Forward Networks without being affected by the vanishing/exploding gradient problem (Sussillo, 2014). RNNs, on the other hand, suffer from the vanishing or exploding gradient problems when trying to "remember" information about past steps which lie many lags back (Hochreiter, 1991). That is, the gradients responsible for the parameter update process tend to, as the distance from the final layer increases, get close to zero, leading to extremely long training times, or explode, causing weight instability during the process.

Given how observations that lie more than one time period into the past can have a strong effect on future outcomes, using these networks for Time Series Forecasting is particularly problematic. We need the ability to save information about the past and at the same time our Network should be able to finish training within a reasonable time frame.

2.4.3 LSTMs

An LSTM (Long Short-Term Memory) Network is a kind of RNN called a gated RNN which is, due to its architecture, particularly well fitted to solving Time Series prediction tasks like the one at hand. It has been reported that LSTMs outperform traditional Time-Series Methods like ARIMA (Autoregressive Integrated Moving Average), improving prediction accuracy by 85% on average when compared to the latter (Siami-Namini et al., 2018). LSTMs solve the vanishing/exploding gradient problems through the use of gate units (Hochreiter and Schmidhuber, 1997). In particular, by integrating so called input and output gates into the architecture as can be seen in Figure 3.

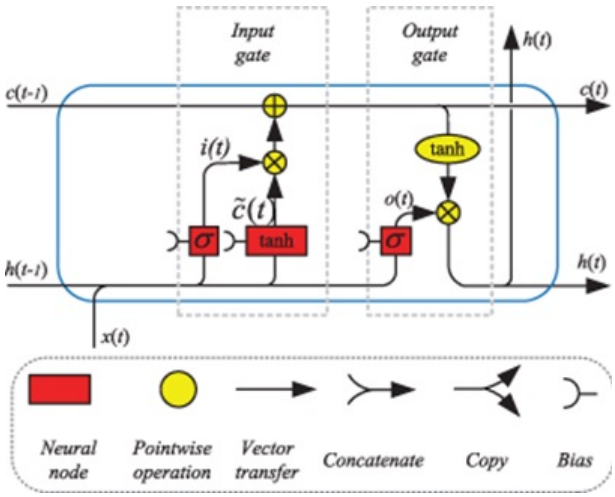

Figure 3: LSTM cell from Yu et al. (2019)

On Figure 3 we can see the workings of an LSTM cell. This can also be represented in mathematical terms as follows (Yu et al., 2019),

$$
\begin{aligned}
i_t &= \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i), \\
\tilde{c}_t &= tanh(W_{\tilde{c}h}h_{t-1} + W_{\tilde{c}x}x_t + b_{\tilde{c}}), \\
c_t &= c_{t-1} + i_t \cdot \tilde{c}_t, \\
o_t &= \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o), \\
h_t &= o_t \cdot tanh(c_t),
\end{aligned}
$$

with $W_i, W_o, W_{\tilde{c}}$ being the weights, c_t the cell state, x_t the inputs, h_t the outputs. " $\cdot$ " stands for point-wise multiplication of vectors. The red squared σ are sigmoid layers. The red rectangle is a *tanh* layer.

There are two layers within the input gate. Firstly, the σ layer controls which information gets through. Secondly, the *tanh* layer computes a new cell state $\tilde{c}_t$. Their outputs are then point-wise multiplied and summed into the previous cell state to update it. All in all, what the gate does is decide which new information will be stored in the cell state.

The output gate, on the other hand, controls which information will be finally outputted taking account of the cell state. The σ within the output gate controls the error flow to the output connections (W_o). The updated cell state coming from the input gate is put through tanh and scales the output of the sigmoid gate.

Hochreiter and Schmidhuber (1997) explicitly reset cell states to zero after each sequence. The cell states, however, often grow linearly, such that if they are not reset, the cell eventually degenerates to a BPTT unit. Sometimes we have no previous knowledge of the typical lag size or of when a sequence should start or end. This is what Gers et al. (2000) realized and solved with the inclusion of forget gates into a traditional LSTM architecture. This modified LSTM architecture can be seen in Figure 4.

As depicted in Figure 4, the input and output gates remain unchanged. Forget gates learn to decide when the information stored in the cell is declining in utility and

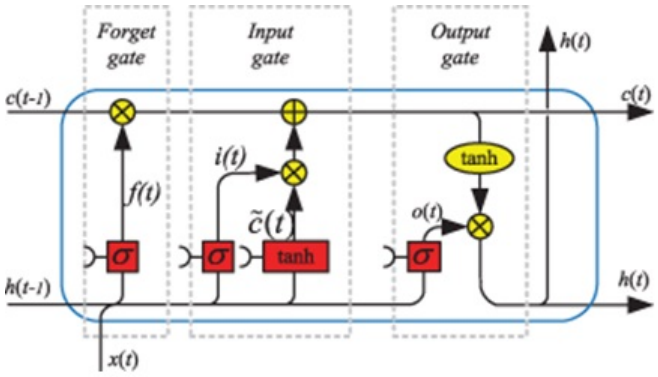

Figure 4: LSTM cell from Yu et al. (2019)

relevance and then proceed to reset the cell state in a gradual or sudden manner (Gers et al., 2000). When referring to an LSTM cell, people are usually making reference to this particular architecture, which can in turn be mathematically expressed as follows (Yu et al., 2019):

$$
\begin{aligned}
f_t &= \sigma(W_{fh}h_{t-1} + W_{fx}x_t + b_f), \\
i_t &= \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i), \\
\tilde{c}_t &= tanh(W_{\tilde{c}h}h_{t-1} + W_{\tilde{c}x}x_t + b_{\tilde{c}}), \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t, \\
o_t &= \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o), \\
h_t &= o_t \cdot tanh(c_t),
\end{aligned}
$$

with f_t the value of the forget gate. If f_t is one, then all information is kept. On the contrary, if it's value is zero, it gets rid of all stored information (Olah, 2015).

2.4.4 GRUs

While LSTMs outperform RNNs in most time-series prediction tasks, they are also more computationally expensive. Cho et al. (2014) proposed an alternative to the LSTM with forget gate called a GRU (Gated Recurrent Unit). What they proposed

was an architecture which integrated the forget and input gate of an LSTM cell into a so called update gate. In figure 5 we can see the structure of a GRU cell, whose mathematical representation is presented next:

$$
\begin{aligned}
r_t &= \sigma(W_{rh}h_{t-1} + W_{rx}x_t + b_r), \\
z_t &= \sigma(W_{zh}h_{t-1} + W_{zx}x_t + b_z), \\
\tilde{h}_t &= tanh(W_{\tilde{h}h}(r_t \cdot h_{t-1}) + W_{\tilde{h}x}x_t + b_z), \\
h_t &= (1 - z_t) \cdot \tilde{h}_t + z_t \cdot h_{t-1},
\end{aligned}
$$

with r_t the reset gate and z_t the update gate.

The reset gate functions as follows. If it is zero, then the previous cell state is ignored and the new one is reset using the current input only. If it were one, the opposite would happen and the inputs would be completely ignored. In this way, the hidden state h_t is able to drop irrelevant information as this starts to become obvious in future time steps. The update gate, on the other hand decides how the previous cell state will impact the current one. The reset and update gates being independent, each cell will capture correlations over different time scales (Cho et al., 2014).

GRUs don't have as much representational power as LSTMs, they are, however, cheaper to train while still having a pretty good performance (Yu et al., 2019). Moreover, the GRU has been shown to outperform traditional RNNs (Chung et al., 2014).

2.4.5 Drop-out

When using complex models such as LSTMs and GRUs in, for machine learning, small training sets, as the one presented in Section 3, we are always under risk of overfitting to our training data. This would lead to a model that apparently performs well, but is just actually learning the data set as a whole. This can be then seen when we test this model on a held-out data set and suddenly it does not perform as well any more.

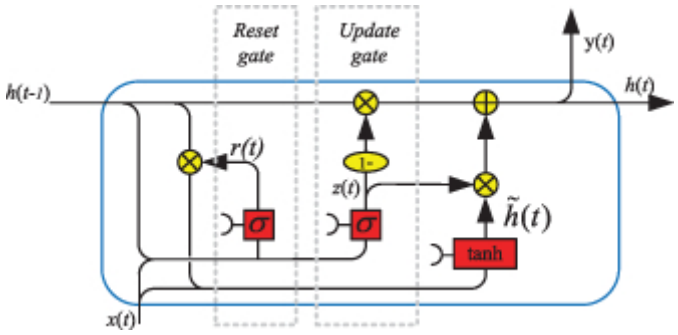

Figure 5: GRU cell from Yu et al. (2019)

One simple way of avoiding overfitting is just to not over do it with the parameters of the model. That is, by not stacking or adding more units to a layer than strictly necessary. Most of the time, however, it is not known how many layers and units per layer would be adequate.

Hinton et al. (2012) propose an alternative to reduce overfitting called Drop-out. What they propose is randomly dropping some of the hidden units before each training case. This stops the network from over relying on sets of hidden units working together and thus preventing highly complex co-adaptations which would en up learning to model the training data. The original Drop-out paper introduced Drop-out for the case of deep Feed-Forward Neural Networks. (Gal, 2016) expanded on this to provide an alternative which also works for RNNs. The innovation consists in applying the same Drop-out mask to each training step in a pass.

3 Data and Implementation

3.1 Data

The data used consists, as of the 21st May 2021, of 2487 observations of the CRIX obtained from `https://thecrix.de/`. The first observation was taken on the 28th of July, 2014 and each new day another observation is added to the series up to the present day. The CRIX Index has, as of 21st May 2021, 10 components which can be seen in Table 1. The prices are always in USD. See Table 2 for an overview of the symbols.

3.1.1 The CRIX

The CRIX (CRyptocurrency IndeX) is based on work done by Trimborn and Härdle (2018). A graph of the evolution of the Index since its inception can be seen in Figure 6. For its construction the adjusted formula of Laspeyres is used, which is itself derived from Laspeyres' Index formula:

$$P_{0t}^{L}(k) = \frac{\sum_{i=1}^{k} P_{it} Q_{i0}}{\sum_{i=1}^{k} P_{i0} Q_{i0}} \tag{3}$$

with $P_{0t}^{L}(k)$ the index level for a basket of k assets on period t with respect to base period 0, P_{it} the Price at time t of a given asset i and Q_{i0} the amount of shares (or in our case coins) of an asset i at time 0 (our base period).

The goal of an index is to be a representative measure of the market. The Laspeyres Index allows us to compare how much the cost of the market basket at prices of period t differ from the cost of the same market basket on the base period 0. However, it has problem dealing with changes in the market composition. An index should only change if there is a price change, not a structure change like a new constituent which becomes part or an old one which drops out.

Figure 6: Graph of the all time Price of the CRIX Index (in USD) as of the 21st of May, 2021

As markets are not monolithic, but fluctuate, and what was representative of a given market at one point in time, might not be representative of the same market at another, it is necessary for the index to change with the market. Companies can go bust or increase their amount of shares, new companies can gain importance as well. Not only should the index be able to change with these kinds of developments, it also should do so in a way which enables comparison between pre and post adjustment levels.

Many important equity indices like, for example, the DAX (Deutsche Boerse AG, 2019) use the adjusted Laspeyres Index to escape this problematic. The Cryptocurrency space is extremely new and unconsolidated. As a result of that, coins gain and lose in importance over short periods of time. This makes the adjusted index of Laspeyres a necessity. Its mathematical representation is the following:

$$CRIX_t(k, \beta) = \frac{\sum_{i=1}^{k} \beta_{i,t_l^-} P_{it} Q_{it_l^-}}{Divisor(k)_{t_l^-}} \tag{4}$$

with β_{i,t_l^-} the adjustment factor for asset i at time point t_l^- (last time point at which $Q_{it_l^-}$, β_{i,t_l^-} and $Divisor(k)_{t_l^-}$ were updated). l tells us that we are dealing with the l-th adjustment factor (Trimborn and Härdle, 2018). The $Divisor(k)_0$ is defined as

$$Divisor(k)_0 = \frac{\sum_{i=1}^{k} \beta_{i0} P_{i0} Q_{i0}}{1000} \tag{5}$$

Whenever the structure of the market, and therefore the components, change, the divisor is adjusted in order to avoid non-price-related movements on the level of the index.

Symbol	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	SD
CRIX	342.10	848.50	10719.40	19462.90	22832.00	173952.80	31506.00
BTC	67.81	425.24	1700.47	6247.23	8225.74	63576.68	10764.01
ETH	0.433	13.315	194.162	339.826	368.545	4182.790	529.026
BNB	0.0398	10.4159	16.2522	44.4099	26.8605	675.0990	106.5288
XRP	0.0027	0.0073	0.1768	0.2255	0.3065	3.3985	0.3325
DOGE	0.0001	0.0002	0.00177	0.0088	0.0028	0.6818	0.0497
USDT	0.5725	0.9998	1.0000	1.0007	1.0009	1.3231	0.0181
ADA	0.0213	0.0458	0.0848	0.2155	0.1641	2.2862	0.3449
DOT	2.872	4.680	8.766	17.629	34.083	47.332	14.766
LTC	1.149	3.795	26.957	47.507	60.548	384.672	62.168
LINK	0.148	0.437	1.892	5.909	4.784	51.852	9.603

Table 1: Descriptive statistics of CRIX's Price and that of its 10 components (in USD) as of the 21st of May, 2021

3.1.2 The components

Figure 7 shows the evolution of BTC and ETH, which put together make around 81.12% of the Index weight according to `https://thecrix.de/`, as of the 21st of May, 2021.

As we can see in the charts for all cryptos there are peaks around the end of 2017 with the exception of DOT (which did not exist back then) and USDT (stable coin designed to always be worth one dollar). This is coherent with the Cryptocurrency boom at the end of 2017 and the following crashing of the market. The peaks in BNB's and LINK's chart are much more less pronounced than the rest, likely because of being new cryptocurrencies at the time. We only have data for a couple months before the crash (16th of September, 2017 for BNB and 9th of November, 2017 for LINK, see Table 2). We also observe a very small peak in comparison to the other Cryptocurrencies for DOGE, possibly linked to its status as a Memecoin and its having been created as a joke (ABC, 2021).

We can also see an incredible growth since October of 2020 in accordance to the

Symbol	Name	Start Date
CRIX	Cryptocurrency Index	2014-07-31
BTC	Bitcoin	2013-04-28
ETH	Ethereum	2015-08-07
BNB	Binance Coin	2017-09-16
XRP	Ripple	2013-08-04
DOGE	Dogecoin	2013-12-15
USDT	Tether	2015-02-25
ADA	Cardano	2017-10-18
DOT	Polkadot	2020-08-19
LTC	Litecoin	2013-04-28
LINK	Chainlink	2017-11-09

Table 2: Symbols, Names and Start Dates of data for CRIX and its 10 components

Crypto Boom we have been experiencing since then. Many all-time-highs were achieved for the components and the CRIX itself in the last 8 months. For an overview of the descriptive statistics of the CRIX and its components see Table 1

3.1.3 The methodology

As our goal was not to predict CRIX's price, but it's log returns, the following transformation was applied to the series:

$$R_j = \frac{Ln(P_j) - Ln(P_{j-t})}{t} \tag{6}$$

where R_j is the log return at day j, P_j is the price of the CRIX at day j and t is the amount of days between the two dates considered (in our case it will be 1, because we want daily log returns).

In Figure 11 the autocorrelation and partial autocorrelation plots for the log returns series can be found. The series seems to be stationary with a significative 6th lag on its ACF. The 1st Lag is not significative.

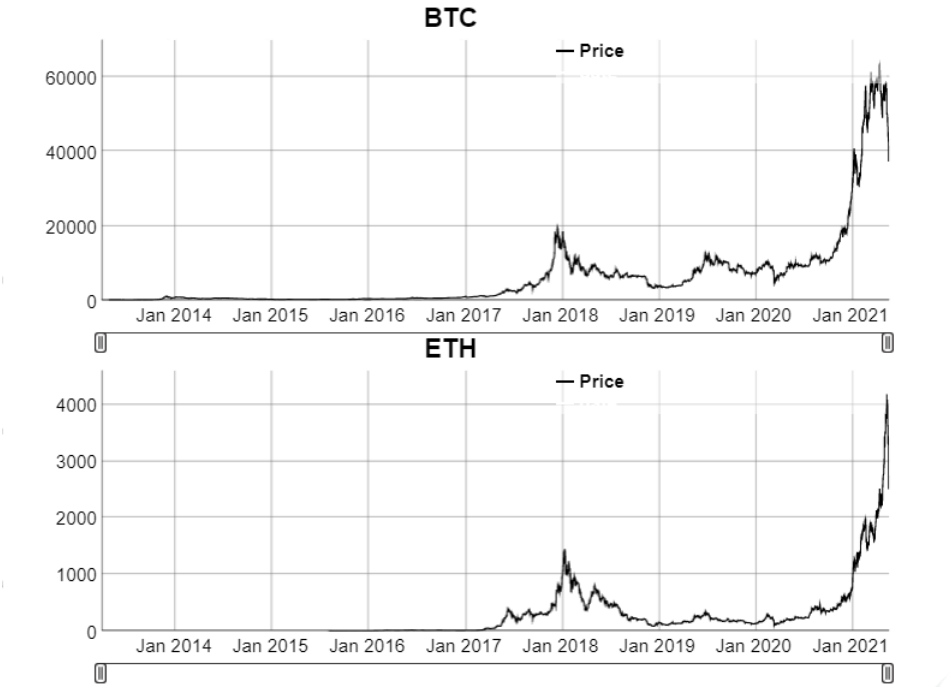

Figure 7: Graph of the Prices of BTC and ETH as of the 21st of May, 2021

Furthermore, a decomposition of the monthly log returns showed no seasonality and no trend on the series as can be seen in Figure 12. The seasonality oscillates in the range of $(-0.01, 0.01)$ and the trend around $(-0.02, 0.01)$, all negligible values at a monthly scale and taking into account the volatility of the assets behind the series. As can also be seen in Figure 12, the Remainder is comparatively big, when contrasted to the seasonal and trend components.

As normal Cross-Validation does not work for Time Series models as it would destroy the temporal component, the "forecast" package uses a special Cross Validation for time series described in Hyndman and Athanasopoulos (2018). Time Series Cross Validation consists in taking just one observation to validate the performance of the model trained on all previous observations with a given forecast horizon (in our case one day into the future). Some of the first observations in the series are not used as validation set, because their use would imply training the model on very few observations. This is graphically presented on Figure 10. For training the Machine Learning Models, the batches of samples where created with generator functions to avoid breaking the temporal aspect of the series.

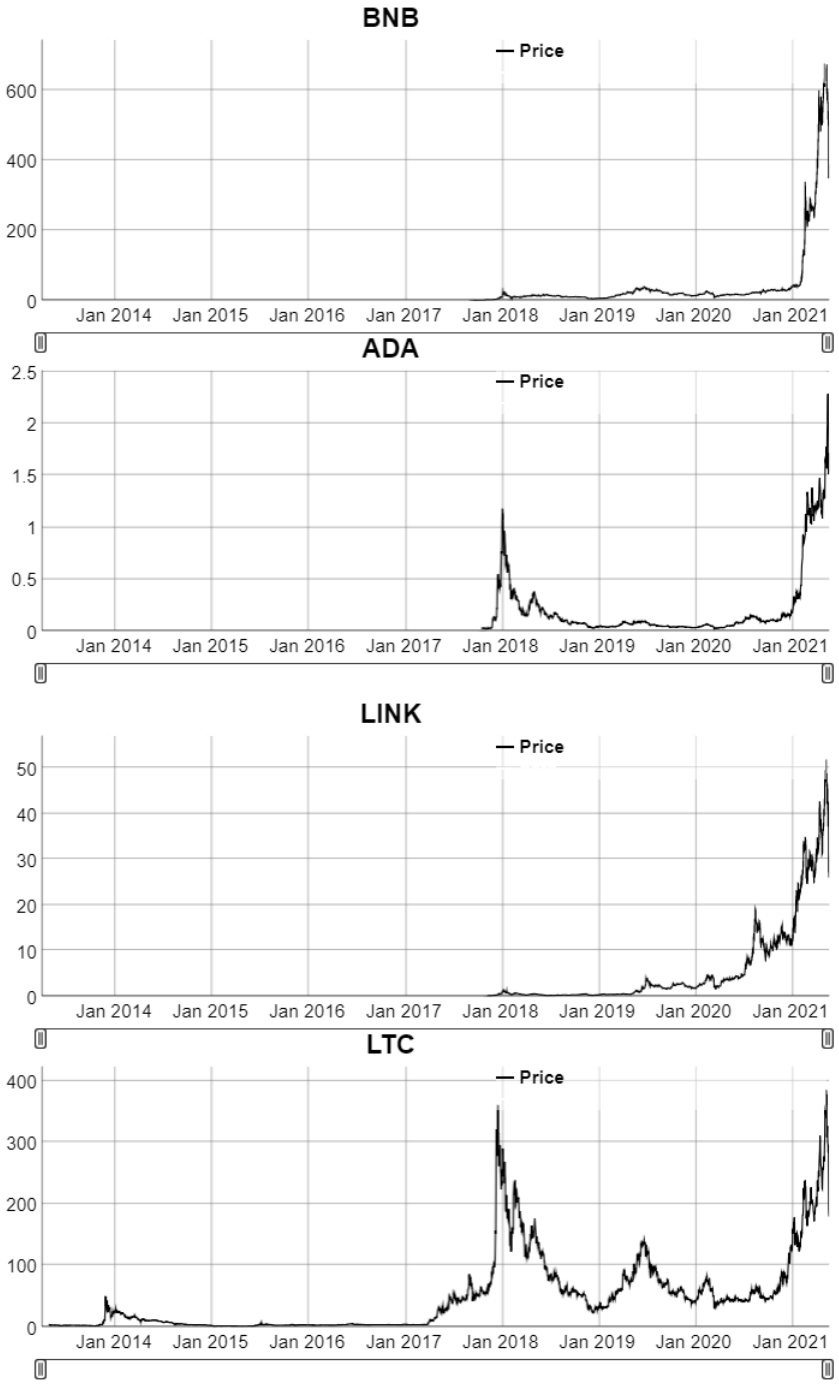

Figure 8: Graphs of the Price for BNB, ADA, LINK and LTC respectively as of the 21st of May, 2021

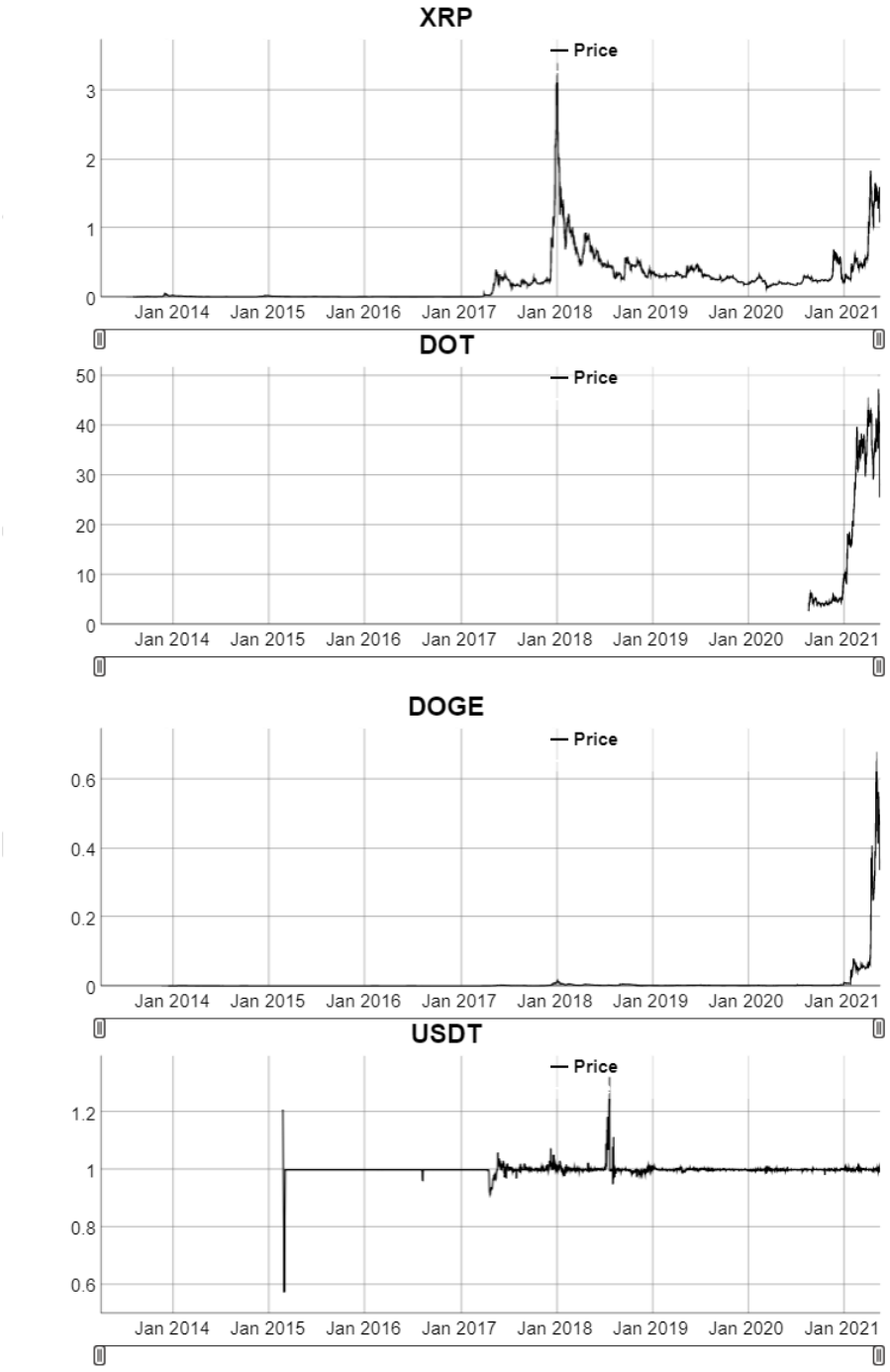

Figure 9: Graph of the Price for XRP, DOT, DOGE and USDT respectively as of the 21st of May, 2021

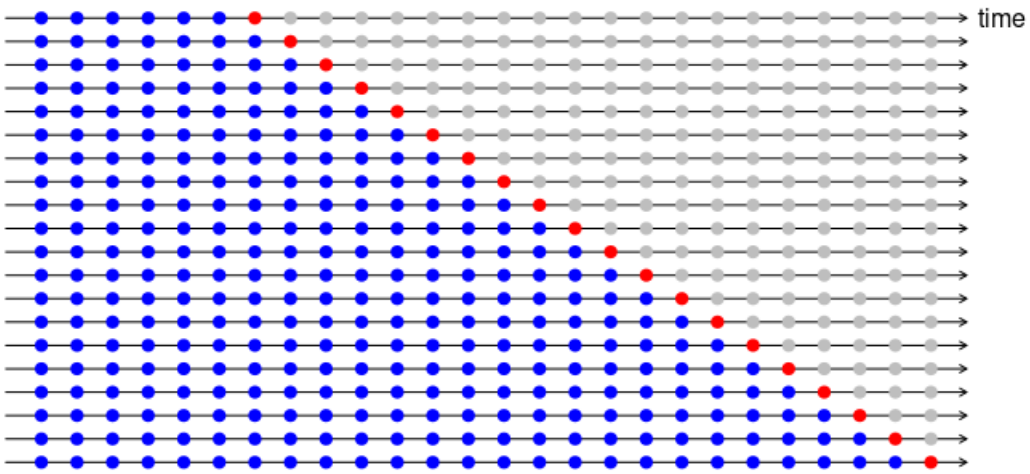

Figure 10: Time Series Cross Validation with observations used for training in blue and red observations used for validation. Each horizontal line from the top represents consecutive training passes. Graph by Hyndman and Athanasopoulos (2018)

For modelling and forecasting the "Keras" (Allaire and Chollet, 2020) and "forecast" (see Hyndman et al. (2020) and Hyndman and Khandakar (2008)) packages were used. The data was divided into training and test sets with Cross-Validation Mean Absolute Error (MAE) being the chosen measure for model selection. The chosen model and the benchmark were then applied to the test set to get the out-of-sample MAE. The code can be found on `https://github.com/QuantLet/CRIXForecastApp`.

3.2 App

The App has been built to be modular and easy to expand, so as to allow improvements and more variety on the models and time series it takes into account. The App was built using "R" (R Core Team, 2013). For the web interface "Shiny" (Chang et al., 2020) as well as "dygraphs" (Vanderkam et al., 2018) were used. To see a full list of the packages as well as the settings used see Appendix A. For a reference on the keras package see Chollet and Allaire (2018) and for the forecast package see Hyndman and Athanasopoulos (2018)

When we open the App we see a horizontal navigation bar on top. There are different pages within the App that are accessible through the navigation bar. These pages are "Forecast", "Analysis", "References", "Terms of Use", "Contact" and "About". We proceed to describe the functioning of the App through each of the individual pages.

3.2.1 The Forecast Page

First, we have the "Forecast" tab. In this page resides most of the functionality of the App and it is this tab that is selected on start-up. On the left we have a selection menu with three basic fields and a "Plot" button. The first of those is the "Entity" field, which is, in accordance with the scope of this work, restricted to the CRIX. It is easy to see, that if we try to select it and input text, nothing really changes.

The second field, as its name indicates, allows us to enter a number which represents the amount of days to forecast into the future. By default it is set to one as this is the forecasting horizon for which the main prediction model (LSTM) was configured and trained. For other models, however it might be helpful to predict more than one period into the future in order to get a longer forecast horizon or to help with visualization. It is not possible to input text on this field and if non integer numbers are given in, the digits after the comma will be ignored (e.g. 2,99999 will be interpreted as 2). It is possible to input negative numbers in which case the App defaults to showing a plot of the evolution of the chosen entity without forecasting.

The "Select Model:" field is a drop down menu which allows us to choose the model which is used for prediction. It is by default set to the "LSTM" model as this is the principal workhorse of the App. The Naive and Mean benchmarks, as well as some other models can also be selected.

After having selected the settings we want to use, we can go ahead an click on the "Plot" button. This will generate an interactive graph of the Log returns of the CRIX over the last 30 days (in black) and any forecast values we have set up (in Blue). If

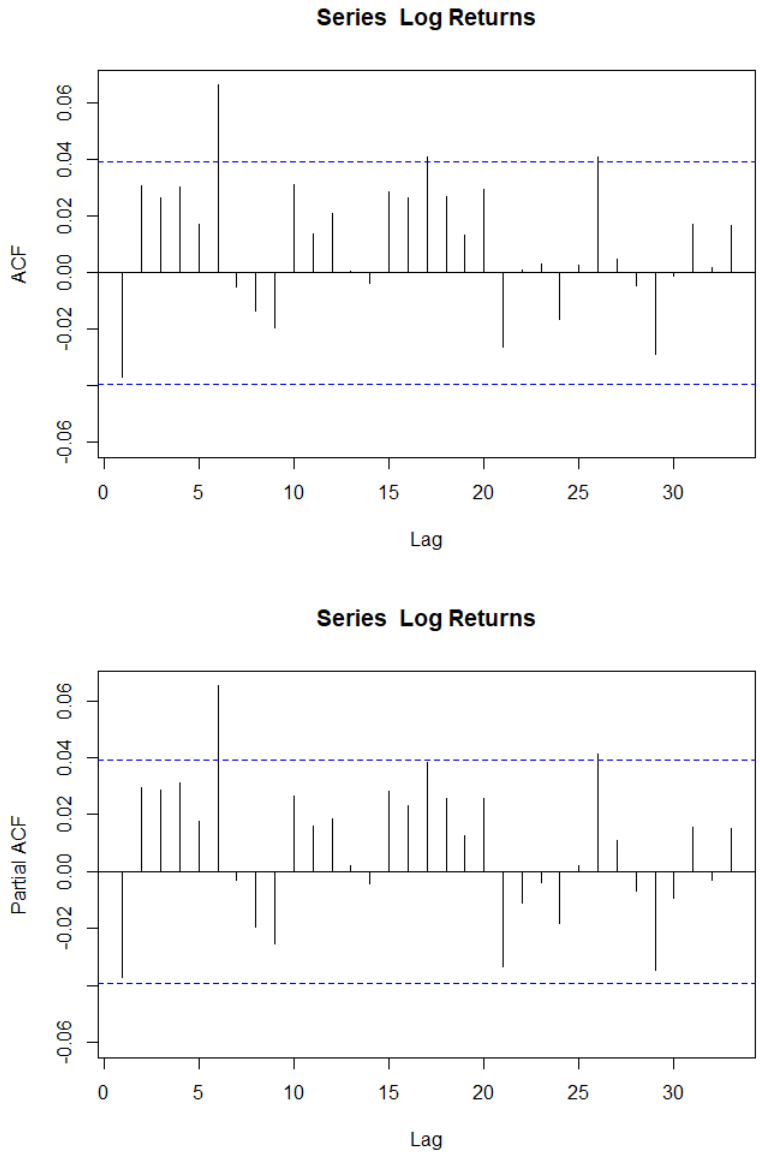

Figure 11: Autocorrelation and Partial Autocorrelation plots for the Log Returns Series for the CRIX

we hover over the graph, a legend will appear with the date of the point over which we hover, as well as the value of the series at that date. In case confidence intervals or any other type of bands are available, these will also be in blue and their value at some point will be displayed in the legend. The point itself will change its size while we hover over it so that it is easily recognizable which is the observation or forecast the legend refers to. We can get a forecasted value by hovering over any of the blue points (or the vertical range of those point on the graph). The observation which has as label "13th of May, 2021" denotes the Log Return between the 12th of May, 2021 as initial date and the 13th of May, 2021 as final date. A range selector can be found under the x-axis which can be used to change the dates which we can see. It is also possible to zoom in on some range by clicking and dragging between the desired dates. It should be noted that each time a setting is changed the "Plot" button has to be clicked again for the graph to be updated.

There is some additional fields which appear conditional on the model selected. In particular, if we select a parametric model (for example "ETS") on the second field, the parameters of that model will be displayed as non modifiable fields between the "Select Model:" field and the "Plot" button.

3.2.2 The Other Pages

In the "Analysis" tab we can see an interactive plot of the monthly decomposition from Figure 12, as well as the Autocorrelation and Partial autocorrelation plots of Figure 11. The "Entity:" Field shows us that the CRIX is selected. Given it is the only entity that can be chosen, nothing can be inputted into that field. The other field, labelled "Select Analysis:" allows us to select which of the monthly decomposition, ACF or PACF should be displayed.

The "References", "Terms of Use", "Contact" and "About" are just placeholders where possible information regarding the App can be documented. None of those are necessary for the functioning of the core features. They can be erased or replaced if the need arises.

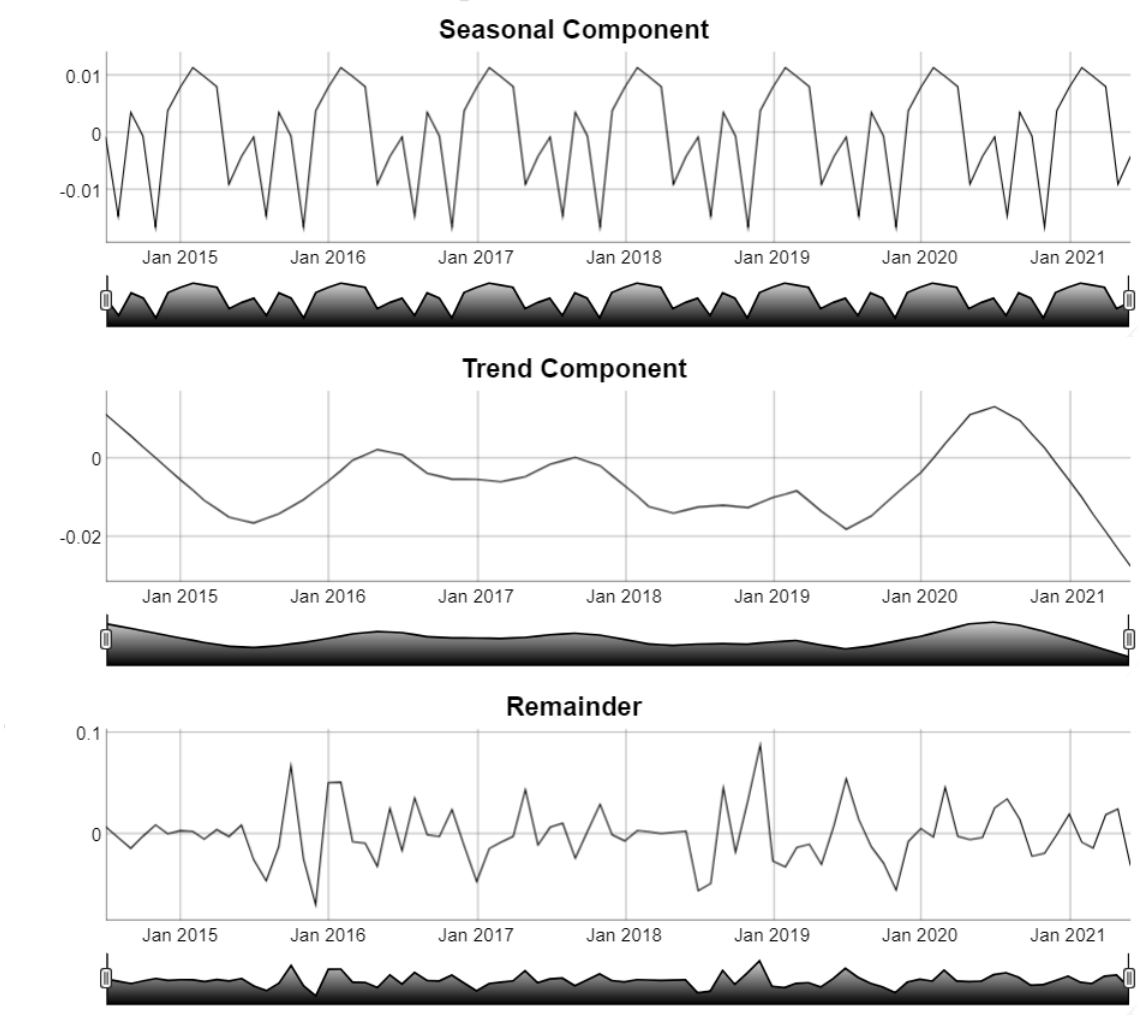

Figure 12: Monthly decomposition of the Log Returns Series for the CRIX

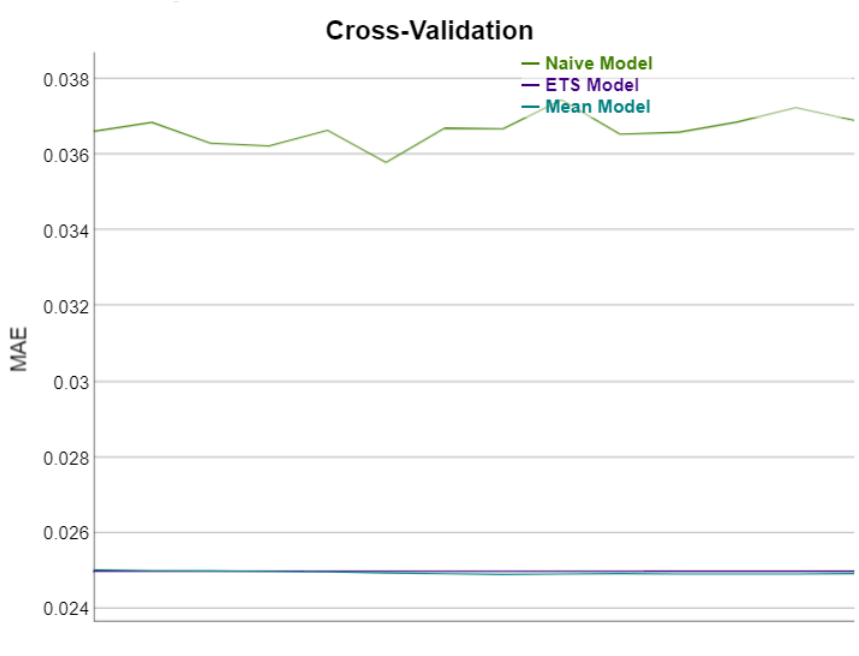

Figure 13: Cross-Validation of Naive, Mean and ETS Models on a forecast horizon of 14 steps into the future

4 Results

First, the performance of the ETS model was compared to the mean and naive forecasts, as an attempt with a simple model before trying with a more complex one. In Figure 13 we see that the ETS Model is just not flexible enough and the best performance tends to lie in just imitating a Mean model, that is, by giving up on the Forecast by defaulting to the most simple alternative and estimating an α which is really small (0.0001). Such a small α implies almost no weight at all is put on observations which came before. The simplest ETS model is the ETS(A,N,N), i.e. simple exponential smoothing with additive errors. It is also the ETS configuration with best performance beating both Naive and Mean baselines. Searching for a more complex model is justified by these results. It is worth noting that all MAEs were calculated as of June 7th, 2021.

Table 3 shows the models and their corresponding MAE. First, a simple non-recurrent machine learning model was tried. This ML Baseline did outperform the Naive model, but failed at improving the Cross-Validation MAE beyond that of the

Mean Benchmark. Three different specifications of the GRU were tried next: a simple GRU model, a GRU model with Recurrent Drop-Out and a model with two GRU layers stacked. However, as before the performance of the Mean benchmark was not achieved. Lastly, two LSTM models were specified, a LSTM with Recurrent Drop-Out and a model which consisted of two LSTM layers stacked with Recurrent Drop-Out being applied to both of them. As we can see all ML models were able to comfortably beat the Naive benchmark, but only the LSTM with Recurrent Drop-Out was able to also beat the Mean benchmark. With this, our model selection was concluded and the LSTM with Recurrent Drop-Out emerged as the victor.

It is however of utmost importance, when training ML models capable of such flexibility, to be suspicious of training results and be aware of the possibility of overfitting. A first measure against it is already integrated into the training process of the model, which was achieved with the help of a Validation set. Another step in that direction is taking a look at the performance of the chosen model on out-of-sample data. With this purpose in mind a set of observations was excluded from the data used in training to serve as a test. Fortunately, the LSTM with Recurrent Drop-Out has a MAE of 0.02441401 on the test set, further confirming our choice. Surprisingly, the model performs even better than in the training set, which might indicate some characteristic of the CRIX changes with time and is making later observations easier to predict with this model. This may warrant further investigation into the causes and nature of that change.

Model	Cross-Validation MAE
Naive	0.03661330
Mean	0.02503500
ETS	0.02499423
ML Baseline	0.02669032
GRU	0.02527644
GRU with Recurrent Drop-Out	0.02502402
GRU Stacked	0.02506776
LSTM Stacked with Recurrent Drop-Out	0.02502208
LSTM with Recurrent Drop-Out	0.02498801

Table 3: Cross-Validation Results

5 Conclusions

The Cryptocurrency market has been in constant flux the last few years and Cryptocurrencies have become an asset which gains more and more recognition and adepts as time goes by. We must strive to comprehend this market as its importance and associated risks grow. The Shiny App which resulted of this thesis should serve both as a display of the capabilities of the model and as a starting platform from which to further improve it. Hopefully, it will be successful on achieving those goals.

The LSTM with Recurrent Drop-Out model presented on this thesis already beats Naive and Mean Benchmarks, and does this only using the time series itself as inputs. It also performs well on out-of-sample data. There is potential to expand this including other variables which could be impacting the Crypto market. The fact that the model performs even better on the test set than on the training set might be an indication that something has changed on the series over time. This would also be worth some investigation.

One promising alternative would be integrating some type of sentiment analysis, given the passion and sometimes craze Cryptos have been provoking over the first few months of 2021, both rather ethereal when it comes to measurements. Another path worthy of study would be looking into each particular component to see if, by

taking a more atomic approach to the analysis, better performance can be achieved. It is my sincere hope that this work contributes to spur more investigation into the Cryptocurrencies market and particularly into the workings of the CRIX.

References

ABC (2021): "What is Dogecoin and what does it have to do with Elon Musk and Saturday Night Live?" https://www.abc.net.au/news/2021-05-10/what-is-dogecoin-elon-musk-saturday-night-live/100128024, accessed on May 27th, 2021.

Allaire, J. and F. Chollet (2020): *keras: R Interface to 'Keras'*, r package version 2.3.0.0.

Badea, L. and C. Mungiu-Pupãzan (2021): "The Economic and Environmental Impact of Bitcoin," *IEEE Access*, PP, 1–1.

Blau, B. M., T. G. Griffith, and R. J. Whitby (2021): "Inflation and Bitcoin: A descriptive time-series analysis," *Economics Letters*, 203, 109848.

Blockchain.com (2021): "Block 0," https://www.blockchain.com/btc/block/0, accessed on June 8th, 2021.

Business Insider (2021): "Facebook's key partners on its cryptocurrency Libra are refusing to publicly support it," https://www.businessinsider.com/libra-association-members-refuse-public-support-facebook-cryptocurrency-plan-2019-8?r=DE&IR=T, accessed on May 18th, 2021.

Buterin, V. (2013): "Ethereum Whitepaper," https://ethereum.org/en/whitepaper/, accessed on May 17th, 2021.

Chang, W., J. Cheng, J. Allaire, Y. Xie, and J. McPherson (2020): *shiny: Web Application Framework for R*, r package version 1.5.0.

Chen, C., W. K. Härdle, A. Hou, and W. Wang (2018): "Pricing Cryptocurrency Options: The Case of CRIX and Bitcoin," *SSRN Electronic Journal*.

Cheng, J., A. N. Lawson, and P. Wong (2021): "Preconditions for a general-purpose central bank digital currency," https://www.federalreserve.gov/econres/notes/feds-notes/

preconditions-for-a-general-purpose-central-bank-digital-currency-20210224.htm, accessed on April 18th, 2021.

Cho, K., B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio (2014): "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," *CoRR*, abs/1406.1078.

Chollet, F. and J. Allaire (2018): *Deep Learning with R*, Manning Publications.

Chung, J., Ç. Gülçehre, K. Cho, and Y. Bengio (2014): "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," *CoRR*, abs/1412.3555.

CoinMarketCap (2021a): "BTC/USD in CoinMarketCap," https://coinmarketcap.com/de/currencies/bitcoin/historical-data/, accessed on June 8th, 2021.

——— (2021b): "ETH/USD in CoinMarketCap," https://coinmarketcap.com/de/currencies/ethereum/historical-data/, accessed on June 8th, 2021.

Deutsche Boerse AG (2019): "Guide to the Equity Indices of Deutsche Boerse AG," Version 9.2.4.

Diem (2019): "Diem White Paper," https://www.diem.com/en-us/white-paper/#cover-letter, accessed on May 18th, 2021.

Diem (2021): "Diem Members," https://www.diem.com/en-us/association/#the_members, accessed on May 18th, 2021.

European Central Bank (2020): *Report on a digital euro*, https://www.ecb.europa.eu/pub/pdf/other/Report_on_a_digital_euro~4d7268b458.en.pdf, accessed on April 18th, 2021.

European Commission and European Central Bank (2021): "Joint statement by the European Commission and the European Central Bank on their cooperation on a digital euro," https://ec.europa.eu/info/sites/default/files/business_economy_euro/banking_and_finance/documents/210119-ec-ecb-joint-statement-digital-euro_en.pdf, accessed on April 18th, 2021.

Gal, Y. (2016): "Uncertainty in Deep Learning," Ph.D. thesis, University of Cambridge.

Gers, F., J. Schmidhuber, and F. Cummins (2000): "Learning to Forget: Continual Prediction with LSTM," *Neural computation*, 12, 2451–71.

Goodfellow, I., Y. Bengio, and A. Courville (2016): *Deep Learning*, MIT Press, `http://www.deeplearningbook.org`.

Hinton, G. E., N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2012): "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, abs/1207.0580.

Hochreiter, J. (1991): "Untersuchungen zu dynamischen Neuronalen Netzen," Diplomarbeit, Technische Universität München.

Hochreiter, J. and J. Schmidhuber (1997): "Long Short-Term Memory," *Neural Computation*, 9, 1735–1780.

Hyndman, R. and G. Athanasopoulos (2018): *Forecasting: principles and practice*, Melbourne, Australia: OTexts, 2 ed.

Hyndman, R., G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay, M. O'Hara-Wild, F. Petropoulos, S. Razbash, E. Wang, and F. Yasmeen (2020): *forecast: Forecasting functions for time series and linear models*, r package version 8.13.

Hyndman, R. J. and Y. Khandakar (2008): "Automatic time series forecasting: the forecast package for R," *Journal of Statistical Software*, 26, 1–22.

Hyndman, R. J., A. B. Koehler, R. D. Snyder, and S. Grose (2002): "A state space framework for automatic forecasting using exponential smoothing methods," *International Journal of Forecasting*, 18, 439–454.

Kim, A., S. Trimborn, and W. Härdle (2019): "VCRIX - A Volatility Index for Crypto-Currencies," *SSRN Electronic Journal*.

Nakamoto, S. (2008): "Bitcoin: A Peer-to-Peer Electronic Cash System," https://bitcoin.org/bitcoin.pdf, accessed on April 15th, 2021.

Olah, C. (2015): "Understanding LSTM Networks," http://colah.github.io/posts/2015-08-Understanding-LSTMs/, accessed on May 17th, 2021.

R Core Team (2013): *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, r version 4.0.2.

Siami-Namini, S., N. Tavakoli, and A. Siami Namin (2018): "A Comparison of ARIMA and LSTM in Forecasting Time Series," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 1394–1401.

Sussillo, D. (2014): "Random Walks: Training Very Deep Nonlinear Feed-Forward Networks with Smart Initialization," *CoRR*, abs/1412.6558.

Sutskever, I., J. Martens, and G. Hinton (2011): "Generating Text with Recurrent Neural Networks," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, Madison, WI, USA: Omnipress, ICML'11, 1017–1024.

Trimborn, S. and W. K. Härdle (2018): "CRIX an Index for cryptocurrencies," *Journal of Empirical Finance*, 49, 107–122.

United Nations (2015): "Paris Agreement," https://unfccc.int/sites/default/files/english_paris_agreement.pdf, accessed on June 8th, 2021.

Vanderkam, D., J. Allaire, J. Owen, D. Gromer, and B. Thieurmel (2018): *dygraphs: Interface to 'Dygraphs' Interactive Time Series Charting Library*, r package version 1.1.1.6.

Vranken, H. (2017): "Sustainability of bitcoin and blockchains," *Current Opinion in Environmental Sustainability*, 28, 1–9.

Werbos, P. (1990): "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, 78, 1550–1560.

Winters, P. R. (1960): "Forecasting Sales by Exponentially Weighted Moving Averages," *Management Science*, 6, 324–342.

Yu, Y., X. Si, C. Hu, and J. Zhang (2019): "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," *Neural Computation*, 31, 1735–1780.

A R Session Info and Packages

- R version 4.0.2 (2020-06-22), `x86_64-w64-mingw32`
- Locale: `LC_COLLATE=German_Germany.1252`, `LC_CTYPE=German_Germany.1252`, `LC_MONETARY=German_Germany.1252`, `LC_NUMERIC=C`, `LC_TIME=German_Germany.1252`
- Running under: `Windows 10 x64 (build 19042)`
- Matrix products: default
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: anomalize 0.2.2, dplyr 1.0.2, dygraphs 1.1.1.6, forcats 0.5.0, forecast 8.13, ggplot2 3.3.2, htmltools 0.5.0, keras 2.3.0.0, lubridate 1.7.9.2, plotly 4.9.2.1, prophet 0.6.1, purrr 0.3.4, Rcpp 1.0.6, readr 1.4.0, reshape2 1.4.4, rlang 0.4.10, shiny 1.5.0, shinyjs 2.0.0, stringr 1.4.0, tibble 3.0.4, tibbletime 0.1.6, tidyr 1.1.2, tidyverse 1.3.0, timetk 2.6.0, xts 0.12.1, zoo 1.8-8
- Loaded via a namespace (and not attached): assertthat 0.2.1, backports 1.2.0, base64enc 0.1-3, broom 0.7.3, cellranger 1.1.0, checkmate 2.0.0, class 7.3-17, cli 2.2.0, codetools 0.2-16, colorspace 2.0-0, compiler 4.0.2, crayon 1.3.4, curl 4.3, data.table 1.13.2, DBI 1.1.0, dbplyr 2.0.0, digest 0.6.27, ellipsis 0.3.1, fansi 0.4.1, fastmap 1.0.1, fastmatch 1.1-0, fracdiff 1.5-1, fs 1.5.0, furrr 0.2.1, future 1.21.0, generics 0.1.0, globals 0.14.0, glue 1.4.2, gower 0.2.2, grid 4.0.2, gtable 0.3.0, haven 2.3.1, hms 0.5.3, htmlwidgets 1.5.2, httpuv 1.5.4, httr 1.4.2, install.load 1.2.3, ipred 0.9-9, jsonlite 1.7.2, later 1.1.0.1, lattice 0.20-41, lava 1.6.8.1, lazyeval 0.2.2, lifecycle 0.2.0, listenv 0.8.0, lmtest 0.9-38, magrittr 2.0.1, MASS 7.3-51.6, Matrix 1.2-18, mime 0.9, modelr 0.1.8, munsell 0.5.0, nlme 3.1-148, nnet 7.3-14, packrat 0.5.0, parallel 4.0.2, parallelly 1.23.0, pillar 1.4.7, pkgconfig 2.0.3, plyr 1.8.6, prodlim 2019.11.13, promises 1.1.1, quadprog 1.5-8, quantmod 0.4.17, R6 2.5.0, readxl 1.3.1, recipes 0.1.15, reprex 0.3.0, reticulate 1.18, rpart 4.1-15, rsample 0.0.8, rsconnect 0.8.16, rstudioapi 0.13, rvest 0.3.6, scales 1.1.1, splines 4.0.2,

stringi 1.5.3, survival 3.1-12, tensorflow 2.2.0, tfruns 1.4, tidyselect 1.1.0, timeDate 3043.102, tools 4.0.2, tseries 0.10-47, TTR 0.24.2, urca 1.3-0, vctrs 0.3.5, viridisLite 0.3.0, whisker 0.4, withr 2.4.1, xml2 1.3.2, xtable 1.8-4, yaml 2.2.1, zeallot 0.1.0

Declaration of Authorship

I hereby confirm that I have authored this Master's thesis independently and without use of others than the indicated sources. All passages which are literally or in general matter taken out of publications or other sources are marked as such.

Berlin, June 22, 2021

Gonzalo Agustin Garcia Camargo